View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Jurnal Eksplora Informatika

IMPLEMENTASI MODUL QUERY GEOGRAFIS OBJEK TITIK UNTUK GOOGLE MAPS API

Ni Made Ika Marini Mandenni
Universitas Udayana
Jalan Raya Bukit Jimbaran, Fakultas Teknik, Jurusan Teknologi Informasi
e-mail: ika_made@yahoo.com

Abstrak

Google Maps API merupakan sebuah layanan yang disediakan oleh pihak Google dalam penyediaan layanan penggunaan peta digital yang terbesar dan terlengkap saat ini. Merupakan suatu antarmuka aplikasi pemrograman yang memungkinkan kita untuk menghubungkan antara aplikasi yang dibuat dengan layanan peta yang dimiliki pihak Google. Namun permasalahan yang sering muncul dengan penggunaan layanan tersebut adalah kurang lengkap atau terbatasnya fitur-fitur query geografis yang disediakan. Sebagai contoh, objek titik yang merupakan objek geografis yang paling sering digunakan dalam pemetaan namun ternyata fitur query geografis untuk objek ini tidak disediakan oleh pihak Google. Hasil akhir dari pembahasan ini adalah menciptakan suatu kelas query geografis untuk objek titik dengan bahasa Javascript, yang akan diintegrasikan dengan kelas-kelas yang sudah disediakan Google Maps API sehingga kesulitan yang paling sering ditemui dalam melakukan query geografis objek-objek titik dalam suatu aplikasi pemetaan digital dapat teratasi.

Kata kunci: *google maps api, kelas query geografis objek titik*

Abstract

Google Maps API is a service provided by the Google in the provision of services use digital maps of the largest and most complete at this time. Is an application programming interface that allows us to connect the applications created with the map service owned by Google. However, problems often arise with the use of such services is incomplete or limited geographic query features are provided. For example, the object is a point which geographic objects are most often used in mapping the geographic queries, but it turned out feature for this object is not provided by the Google. The end result of this discussion is create a class of geographic query language for object point with Javascript, which will be integrated with the classes that have been provided by Google Maps API so that the most frequent difficulties encountered in performing the query objects geographical point in a digital mapping applications can be resolved.

Keywords*: google maps api , geographic query class object point*

1. Pendahuluan

Sistem informasi pemetaan saat ini berkembang begitu pesat. Dukungan teknologi perangkat keras baik komputer server maupun jaringan komputer dengan bandwidth yang besar, memungkinkan pengolahan data berbasis pemetaan geografis berbasis web menjadi lebih cepat dari sebelumnya. Dukungan penyediaan dan pemakaian peta-peta digital yang dilengkapi dengan peta roadmap dan peta citra satelit secara gratis untuk aplikasi-aplikasi pemetaan yang dibuat menyebabkan proses penciptaan aplikasi menjadi lebih cepat dan mudah. Peta-peta digital yang lengkap tersebut biasanya disediakan oleh google maps, yahoo maps maupun microsoft maps secara gratis dalam kondisi penggunaan tertentu.

Google Maps API merupakan sebuah layanan yang disediakan oleh pihak Google dalam penyediaan layanan penggunaan peta digital yang terbesar dan terlengkap saat ini. Merupakan suatu antarmuka aplikasi pemrograman yang memungkinkan kita untuk menghubungkan antara aplikasi yang dibuat dengan layanan peta yang dimiliki pihak Google.

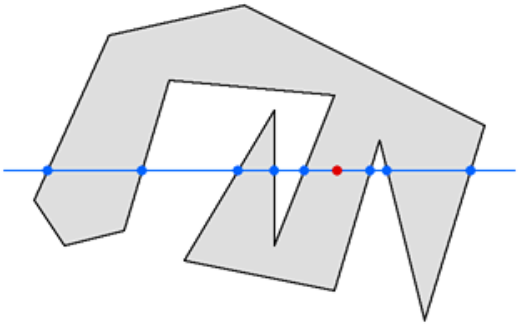
Sistem Informasi Geografis (SIG) adalah sistem informasi khusus yang mengelola data yang memiliki informasi spasial (berbasis keruangan) [3]. Dalam artian yang lebih sempit adalah sistem komputer yang memiliki kemampuan untuk membangun, menyimpan, mengelola dan menampilkan informasi berbasis geografis, misalnya data yang diidentifikasi menurut lokasinya, dalam sebuah database.

Dalam proses pengelolaan data geografis selain membutuhkan permintaan data atau *query* yang berasal dari data tabular atau non spasial, namun juga membutuhkan permintaan data yang berasal dari data spasial yang berbasis keruangan. Namun permasalahan yang sering muncul dengan penggunaan layanan penggunaan peta geografis seperti *Google Maps* adalah kurang lengkap atau terbatasnya fitur-fitur query geografis yang disediakan. Sebagai contoh, objek titik yang merupakan objek geografis yang paling sering digunakan dalam pemetaan namun ternyata fitur query geografis untuk objek ini tidak disediakan oleh pihak Google.

2. Metodologi Penelitian

Dalam bab ini dijelaskan tentang pembahasan dari sistem serta pengujian metode untuk menetukan posisi suatu objek titik terhadap suatu daerah/region yang dibatasi dengan garis-garis polygon yang nantinya akan diimplementasikan kedalam kelas-kelas javascript dan akan dicoba diintegrasikan dengan Google Maps API [3].

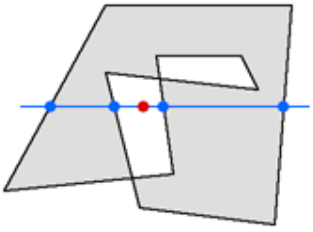
2.1. Kasus ke-1 dalam penyelesaian penentuan lokasi suatu objek titik terhadap suatu bidang.



Gambar 1. Contoh kasus ke-1 dalam pengenalan lokasi titik terhadap suatu bidang/poligon

Solusi yang dapat dilakukan pada permasalahan seperti Gambar 1 dapat dilakukan dengan membandingkan setiap sisi poligon ke koordinat Y (vertikal) dari titik uji, dan menyusun daftar node, dimana setiap node adalah titik di mana satu sisi melintasi ambang Y dari titik uji. Dalam contoh ini, delapan sisi poligon melewati ambang batas Y, sedangkan enam sisi lain tidak. Kemudian, jika jumlah node adalah ganjil pada setiap sisi dari titik uji, maka titik tersebut berada didalam poligon, sedangkan jika ada jumlah node adalah genap pada setiap sisi dari titik uji, maka titik itu berada diluar poligon. Dalam contoh pada Gambar 2.1, terdapat lima buah node di sebelah kiri titik uji, dan terdapat tiga buah node di sebelah kanan titk uji. Karena lima dan tiga adalah bilangan ganjil, maka titik uji dipastikan berada didalam poligon.

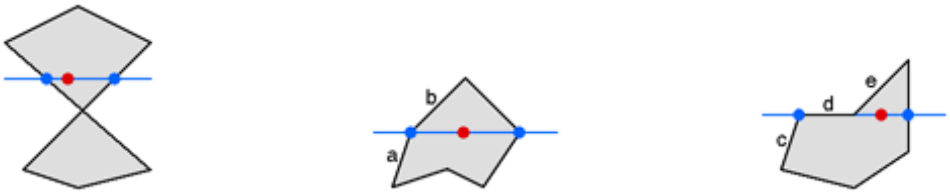
2.2. Kasus ke-2 dalam penyelesaian penentuan lokasi suatu objek titik terhadap suatu bidang.



Gambar 2. Contoh kasus ke-2 dalam pengenalan lokasi titik terhadap suatu bidang/poligon

Gambar 2 menunjukkan apa yang terjadi jika poligon melintasi dirinya sendiri. Pada contoh ini, poligon dengan sepuluh sisi memiliki garis yang saling silang. Efeknya seperti "*eksklusif or*" atau XOR seperti yang dikenal pada bahasa pemrograman assembly. Bagian bangun ruang yang tumpang tindih membatalkan satu sama lain. Jadi, titik uji berada diluar poligon.

2.3. Kasus ke-3 dalam penyelesaian penentuan lokasi suatu objek titik terhadap suatu bidang.



(a) (b) (c)

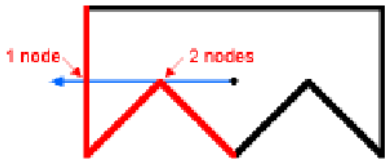
Gambar 3. Contoh kasus ke-3 dalam pengenalan lokasi titik terhadap suatu bidang/poligon

Pada Gambar 3.a, enam sisi poligon tidak tumpang tindih, tetapi memiliki garis- silang. Pada permasalahan ini algoritma masih berfungsi dengan baik.

Gambar 3.b menunjukkan masalah yang terjadi ketika sebuah sudut dari poligon jatuh langsung pada ambang Y. Karena sisi a dan b keduanya bersentuhan langsung dengan titik uji, pertanyaannya apakah akan menghasilkan sebuah node? Jawabannya tidak, akan ada dua node di setiap sisi titik uji sehingga pengujian akan mengatakan bahwa titik berada diluar poligon, padahal kenyataannya tidak. Solusi untuk situasi ini adalah sederhana. Titik yang persis berada di ambang Y dianggap dimiliki oleh satu sisi poligon.

Gambar 3.c menunjukkan kasus poligon di mana salah satu sisinya terletak sepenuhnya di ambang Y dari titik uji. Dengan mengikuti aturan seperti Gambar 3.b sisi c akan menghasilkan sebuah node, karena memiliki satu titik akhir di bawah ambang batas, dan titik akhir lainnya di bawah atau di atas ambang batas. Sisi d tidak menghasilkan node, karena memiliki kedua endpoint atau berada di atas ambang batas. Dan sisi e juga tidak menghasilkan node, karena memiliki kedua endpoint atau berada di atas ambang batas.

2.4. Kasus ke-4 dalam penyelesaian penentuan lokasi suatu objek titik terhadap suatu bidang.



Gambar 4. Contoh kasus ke-4 dalam pengenalan lokasi titik terhadap suatu bidang/poligon

Gambar 2.4 mengilustrasikan kasus khusus dibawa ke perhatian saya oleh John David Munch dari Cal Poly. Salah satu sudut interior poligon hanya menyentuh Y-ambang titik uji. Pada gambar 2.4, hanya satu sisi (berwarna merah) menghasilkan node di sebelah kiri titik uji, dan dalam contoh bawah, tiga sisi dilakukan. Cara lain, jumlahnya ganjil, dan titik uji akan dianggap di dalam polygon.

2.5. Titik-titik Polygon.

Jika titik uji di perbatasan poligon, algoritma ini akan memberikan hasil yang tak terduga, yaitu hasilnya mungkin "dalam" atau "luar" tergantung pada faktor-faktor sewenang-wenang seperti bagaimana poligon berorientasi sehubungan dengan sistem koordinat. (Itu umumnya tidak masalah, karena tepi poligon ini tidak terhingga begitu tipis, dan titik yang jatuh tepat di tepi bisa jalan baik tanpa menyakiti tampilan poligon)

2.6. Implementasi penyelesaian terhadap kasus-kasus pengenalan lokasi titik terhadap suatu bidang/poligon dalam bentuk algoritma pemrograman, berikut beberapa algoritma yang digunakan.

a. Secara umum kita dapat mengimplementasikan penyelesaian terhadap kasus-kasus yang telah dibahas sebelumnya dengan algoritma program [1], [2] sebagai berikut.

```
// Globals yang harus ditetapkan sebelum memanggil fungsi ini:
//
// Int polySides = berapa banyak sudut poligon memiliki
// Mengapung polyX [] = koordinat horizontal sudut
// Mengapung Polyy [] = koordinat vertikal sudut
// Mengapung x, y = titik yang akan diuji
//
// (Globals digunakan dalam contoh ini untuk tujuan kecepatan. Ganti sebagai
// Yang diinginkan.)
//
// Fungsi ini akan mengembalikan YES jika titik x, y berada di dalam poligon, atau
// NO jika tidak. Jika intinya adalah persis di tepi poligon,
// Maka fungsi dapat kembali YA atau TIDAK.
//
// Perhatikan bahwa pembagian dengan nol dihindari karena divisi dilindungi
// Dengan klausa "jika" yang mengelilinginya.


bool pointInPolygon() {

        int   i, j=polySides-1 ;

        bool  oddNodes=NO     ;

        for (i=0; i<polySides; i++) {

          if ((polyY[i]< y && polyY[j]>=y

          ||   polyY[j]< y && polyY[i]>=y)

          &&  (polyX[i]<=x || polyX[j]<=x)) {

            if ( polyX[i]+(y-polyY[i])/(polyY[j]-

                        polyY[i])*(polyX[j]-polyX[i])<x) {

                            oddNodes=!oddNodes;

                }

          }

          j=i;

        }

        return oddNodes;

}
```

b. Dengan menggunakan algoritma dibuat oleh Patrick Mullen, kita juga dapat mengimplementasikan penyelesaian terhadap kasus kasus yang telah dibahas sebelumnya. Algoritma ini berguna jika kita memiliki banyak poin yang perlu diuji terhadap yang sama (statis) polygon.

```
// Globals yang harus ditetapkan sebelum memanggil fungsi-fungsi ini :
//
// Int polySides = berapa banyak sudut poligon memiliki
// Mengapung polyX [ ] = koordinat horizontal sudut
// Mengapung Polyy [ ] = koordinat vertikal sudut
// Mengapung x , y = titik yang akan diuji
//
// Array global menyusul harus dialokasikan sebelum memanggil fungsi-fungsi ini :
//
// Mengapung konstan [ ] = storage untuk konstanta precalculated ( ukuran yang
sama seperti polyX )
// Mengapung beberapa [ ] = penyimpanan untuk pengganda precalculated ( ukuran
yang sama seperti polyX )
//
// ( Globals digunakan dalam contoh ini untuk tujuan kecepatan . Ganti sebagai
// Yang diinginkan . )
//
// PEMAKAIAN :
// Panggil precalc_values ( ) untuk menginisialisasi konstanta [ ] dan beberapa [ ]
array ,
// Kemudian memanggil pointInPolygon ( x , y ) untuk menentukan apakah titik
dalam poligon .
//
// Fungsi ini akan mengembalikan YES jika titik x , y berada di dalam poligon , atau
// NO jika tidak . Jika intinya adalah persis di tepi poligon ,
// Maka fungsi dapat kembali YA atau TIDAK .
//
// Perhatikan bahwa pembagian dengan nol dihindari karena divisi dilindungi
// Dengan klausa "jika" yang mengelilinginya .

void precalc_values() {

        int   i, j=polySides-1 ;

        for(i=0; i<polySides; i++) {

                if(polyY[j]==polyY[i]) {

                constant[i]=polyX[i];

                multiple[i]=0; }

                else {

                constant[i]=polyX[i]-

                        (polyY[i]*polyX[j])/(polyY[j]-

                        polyY[i])+(polyY[i]*polyX[i])

                        /(polyY[j]-polyY[i]);


                        multiple[i]=(polyX[j]-polyX[i])/(polyY[j]-

                        polyY[i]);

                }
```

```
                j=i;
        }
}

bool pointInPolygon() {
        int   i, j=polySides-1 ;
        bool  oddNodes=NO     ;
        for (i=0; i<polySides; i++) {
                if ((polyY[i]< y && polyY[j]>=y
                ||   polyY[j]< y && polyY[i]>=y)) {
                oddNodes^=(y*multiple[i]+constant[i]<x);
                }
                j=i;
        }
        return oddNodes;
}
```

2.7. Pembuatan kelas javascript untuk implementasi query geografis objek titik terhadap suatu bidang geografis [1], [2].

Dengan menggabungkan beberapa algoritma program penyelesaian terhadap kasus-kasus pengenalan lokasi titik terhadap suatu bidang/poligon maka kita dapat membentuk kelas javascript yang diintegrasikan dengan Google Maps API seperti dalam program [1], [2], [3] sebagai berikut.

```
// Polygon getBounds extension - google-maps-extensions
// http://code.google.com/p/google-maps-
// extensions/source/browse/google.maps.Polygon.getBounds.js
if (!google.maps.Polygon.prototype.getBounds) {
        google.maps.Polygon.prototype.getBounds = function(latLng) {
          var bounds = new google.maps.LatLngBounds();
          var paths = this.getPaths();
          var path;

          for (var p = 0; p < paths.getLength(); p++) {
```

```
                path = paths.getAt(p);
                for (var i = 0; i < path.getLength(); i++) {
                  bounds.extend(path.getAt(i));
                }
              }

              return bounds;
            }
}

// Polygon containsLatLng - method to determine if a latLng is
//within a polygon
google.maps.Polygon.prototype.containsLatLng = function(latLng) {
            // Exclude points outside of bounds as there is no way they
            //are in the poly
             var lat, lng;
            //arguments are a pair of lat, lng variables
            if(arguments.length == 2) {
               if(typeof arguments[0]=="number" && typeof
                        arguments[1]=="number") {
                        lat = arguments[0];
                        lng = arguments[1];
               }
            } else if (arguments.length == 1) {
               var bounds = this.getBounds();
               if(bounds != null && !bounds.contains(latLng)) {
                        return false;
               }
               lat = latLng.lat();
               lng = latLng.lng();
             } else {
```

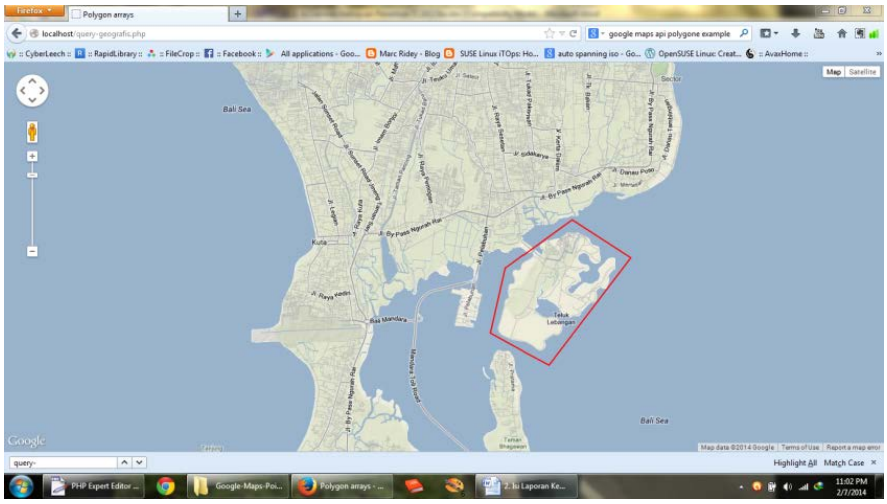
```
        console.log("Wrong number of inputs in
                google.maps.Polygon.prototype.contains.LatLng");
      }
      // Raycast point in polygon method
      var inPoly = false;
      var numPaths = this.getPaths().getLength();
      for(var p = 0; p < numPaths; p++) {
        var path = this.getPaths().getAt(p);
        var numPoints = path.getLength();
        var j = numPoints-1;
        for(var i=0; i < numPoints; i++) {
          var vertex1 = path.getAt(i);
          var vertex2 = path.getAt(j);
          if (vertex1.lng() < lng && vertex2.lng() >= lng ||
                vertex2.lng() < lng && vertex1.lng() >= lng) {
            if (vertex1.lat() + (lng - vertex1.lng()) /
                        (vertex2.lng() - vertex1.lng()) * (vertex2.lat()
                        - vertex1.lat()) < lat) {
                        inPoly = !inPoly;
            }
          }
          j = i;
        }
      }

      return inPoly;
}
```

3. Hasil dan Analisis

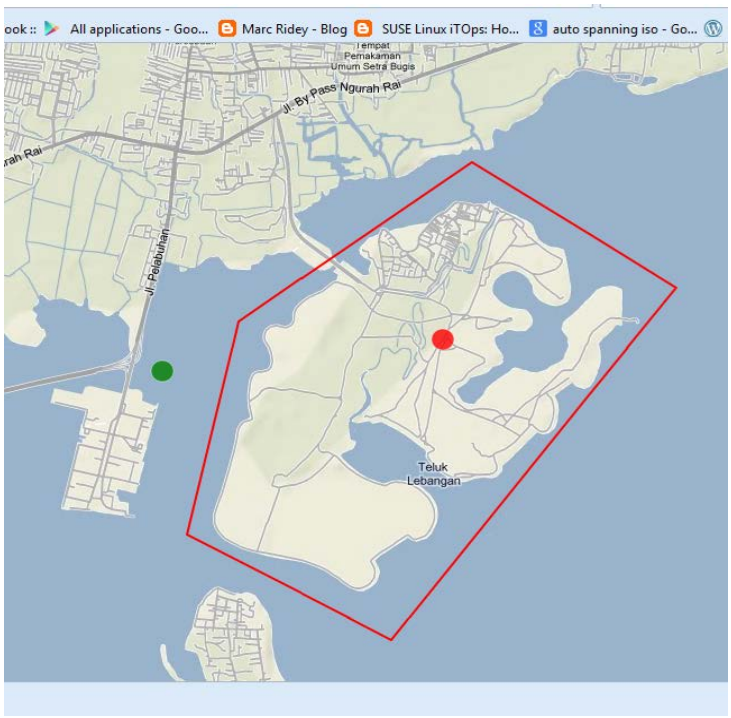
Untuk menentukan apakah algoritma dan kelas javascript yang dibuat berhasil diimplementasikan, maka dibuatlah sebuah program sederhana unutk menguji keberhasilan algoritma tersebut. Sebuah program aplikasi geografis berbasis web dengan menggunakan Google Maps API dibuat dengan

menampilkan peta di daerah Denpasar selatan. Kemudian dibuat sebuah bidang atau region yang membatasi Pulau Serangan sebagai kasus suatu objek area seperti gambar 5.



Gambar 5.Contoh pengujian dengan aplikasi geografis

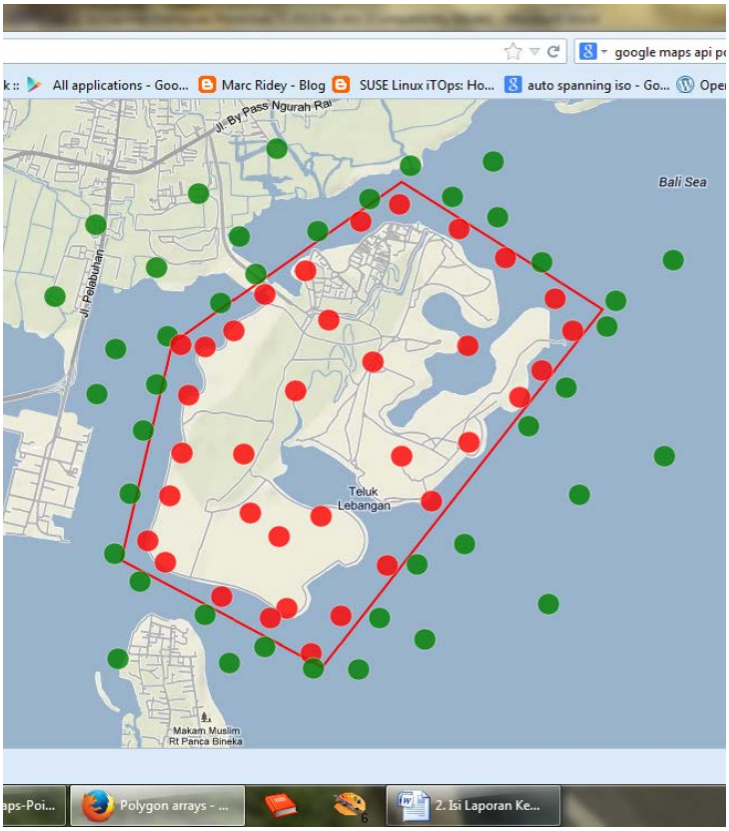
Pada peta, Pulau serangan dibatasi dengan garis merah yang akan dijadikan kasus sebuah area. Kemudian program diuji dengan menempatkan objek-objek titik diluar dan didalam daerah Pulau Serangan tersebut.



Gambar 6.Contoh pengujian objek titik

Objek titik yang berwarna merah melambangkan bahwa objek tersebut berada didalam daerah Pulau Serangan (area yang diuji) dan objek titik yang berwarna hijau menandakan bahwa objek tersebut berada diluar daerah Pulau Serangan.

Untuk melihat keberhasilan dari algoritma dan program yang dibuat, maka ditempatkan beberapa titik didalam dan diluar area yang dimaksud.



Gambar 3.3.Contoh pengujian dengan objek titik yang lebih banyak

Dari gambar 3.3 terlihat pengujian dengan menempatkan lebih banyak objek-objek titik didalam maupun diluar area uji (Pulau Serangan) menunjukkan bahwa algoritma ini berhasil menentukan apakah suatu objek berada didalam ataukah diluar suatu area.

4. Kesimpulan

Adapun kesimpulan yang dapat diambil dalam pembuatan kelas Javascript untuk implementasi modul query geografis titik ini adalah :

1. Dengan kelas javascript yang dibuat dapat digunakan untuk melakukan query geografis terhadap objek titik untuk menentukan posisinya terhadap suatu bidang atau daerah..
2. Dengan adanya kelas-kelas ini akan memudahkan kita dalam melakukan query geografis yang melibatkan objek titik sehingga sangat memudahkan kita dalam programming sistem informasi geografis dengan menggunakan Google Maps API.

Daftar Pustaka

[1] Douglas, C (2012). “JavaScript: The Good Parts”, O'Reilly Media; 1st edition (May 2008)

[2] Matt Weisfeld (2008) “The Object-Oriented Thought Process”, Addison-Wesley Professional; 3 edition (September 4, 2008)

[3] Paul A. Longley “Geographic Information Systems and Science”. Wiley; 3 edition (August 9, 2010).

_____ http://code.google.com, diakses bulan Maret 2013.

_____ http://id.wikipedia.org, diakses bulan Maret 2013.

_____ http://maps.google.com, diakses bulan Maret 2013.